Claims

What is claimed is:

1. A method of editing a software program in a common language runtime environment, the software program having a source code component, an intermediate language component, and a native code component, the method comprising:

executing a first portion of the native code component;

suspending execution of the native code component at a first point;

allowing a user to edit the source code component to create an edited source code component;

compiling the edited source code component using a source compiler to create an edited intermediate language component;

compiling the edited intermediate language component using an intermediate language compiler to create an edited native code component; and

executing the edited native code component beginning at the first point.

2. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to add a new field to an object of a class.

3. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to add a new field to a class.

4. The method of claim 3, wherein allowing a user to edit the source code component comprises allowing the user to add a new field to a class where at least one instance of the class has already been created.

5. The method of claim 2, wherein compiling the edited intermediate language component comprises extending a field table associated with the class to accommodate the new field to create an extended field table, and wherein executing the edited native code component comprises employing the extended field table.

6. The method of claim 5, wherein extending the field table comprises adding the new field via a synchronization block associated with the object to create an extended object.

7. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to add a new method to a class.

8. The method of claim 7, wherein the new method is a virtual method.

9. The method of claim 8, wherein compiling the edited intermediate language component comprises extending a virtual method table associated with the program to accommodate the new method to create an extended virtual method table, and wherein executing the edited native code component comprises employing the extended virtual method table.

10. The method of claim 9, wherein the extended virtual method table comprises first and second memory portions, and wherein extending the virtual method table comprises providing a reference to the new method in the second memory portion.

11. The method of claim 10, wherein the first and second memory portions of the extended virtual method table are non-contiguous, and wherein extending the virtual method table comprises creating a call to the new method using a reference to the second memory portion.

12. The method of claim 11, wherein the second memory portion is located so as to maximize a slot index.

13. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to change an existing method in the source code

component.

14. The method of claim 13, wherein allowing the user to change an existing method comprises allowing the user to add a new variable to the existing method.

15. The method of claim 14, wherein allowing the user to change an existing method comprises allowing the user to add a new variable to the existing method when the existing method is on at least one thread call stack.

16. The method of claim 12, wherein executing the edited native code component comprises substituting edited native code corresponding to the existing method upon a return to the method.

17. The method of claim 16, further comprising determining a return to the method using a breakpoint.

18. The method of claim 13, wherein allowing the user to change an existing method comprises allowing the user to change an algorithm in the existing method.

19. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to replace an existing method created in a first source language with a new method created in a second source language.

20. The method of claim 1, wherein allowing a user to edit the source code component comprises allowing the user to change a first source component associated with a first native component to create an edited first source component, wherein compiling the edited source code component comprises compiling the edited first source component using the source compiler to create an edited first intermediate language component, wherein compiling the edited intermediate language component comprises compiling the edited first intermediate language component using the intermediate

language compiler to create an edited first native component, and wherein executing the edited native code component beginning at the first point comprises executing the edited first native component.

21. The method of claim 1, wherein compiling the edited source code component using the source compiler, compiling the edited intermediate language component, and executing the edited native code component are done on an as-needed basis.

22. A method of editing a software program having a source code component, and a native code component, the method comprising:

executing a first portion of the native code component;

suspending execution of the native code component at a first point;

allowing a user to edit the source code component to create an edited source code component;

converting the edited source code component to create an edited native code component; and

executing the edited native code component beginning at the first point;

wherein allowing a user to edit the source code component comprises allowing the user to perform one of adding a new field to an object of a class, adding a new method to a class, changing an existing method in the source code component, and replacing an existing method created in a first source language with a new method created in a second source language.

23. The method of claim 22, wherein adding a new method to a class comprises adding a new virtual method to the class.

24. The method of claim 23, wherein converting the edited source code component comprises extending a virtual method table associated with the program to accommodate the new virtual method to create an extended virtual method table, and

wherein executing the edited native code component comprises employing the extended method table.

25. The method of claim 24, wherein the extended virtual method table comprises first and second memory portions, and wherein extending the virtual method table comprises providing a reference to the new virtual method in the second memory portion.

26. The method of claim 25, wherein the first and second memory portions of the extended virtual method table are non-contiguous, and wherein extending the virtual method table comprises creating a call to the new virtual method using a reference to the second memory portion.

27. The method of claim 26, wherein creating a call to the new virtual method comprises computing an offset from the first memory portion to the second memory portion.

28. The method of claim 27, wherein the second memory portion is located so as to maximize a slot index.

29. The method of claim 22, wherein changing an existing method comprises adding a new variable to the existing method.

30. The method of claim 29, wherein executing the edited native code component comprises substituting edited native code corresponding to the existing method upon a return to the method.

31. The method of claim 30, further comprising determining a return to the method using a breakpoint.

32. The method of claim 22, wherein changing an existing method comprises changing an algorithm in the existing method.

33. The method of claim 32, wherein allowing the user to change an existing method comprises allowing the user to add a new variable to the existing method when the existing method is on at least one thread call stack.

34. The method of claim 22, wherein allowing a user to edit the source code component comprises allowing the user to perform adding a new field to an object of a class, wherein converting the edited source code component comprises extending a field table associated with the class to accommodate the new field to create an extended field table, and wherein executing the edited native code component comprises employing the extended field table.

35. The method of claim 34, wherein extending the field table comprises adding the new field via a synchronization block associated with the object to create an extended object.

36. A runtime system for executing a program in a computer system, comprising:

an edit and continue component having a debugging services interface component interfacing with a debugger application; and

an intermediate language compiler adapted to compile intermediate language code into native code;

wherein the edit and continue component executes a first portion of a native code component, suspends execution of the native code component at a first point, allows a user to edit the source code component using a debugger application to create an edited source code component, compiles the edited source code component using a source compiler to create an edited intermediate language component, compiles the edited intermediate language component using the intermediate language compiler to create an edited native code component, and executes the edited native code component beginning at the first point.

37. The system of claim 36, wherein the edit and continue component allows a user to add a new field to an object of a class.

38. The system of claim 37, wherein the edit and continue component extends a field table associated with the class to accommodate the new field to create an extended field table.

39. The system of claim 38, wherein the edit and continue component adds the new field via a synchronization block associated with the object to create the extended object.

40. The system of claim 36, wherein the edit and continue component allows a user to add a new method to a class.

41. The system of claim 40, wherein the new method is a virtual method.

42. The system of claim 41, wherein the edit and continue component extends a virtual method table associated with the program to accommodate the new method to create an extended virtual method table.

43. The system of claim 42, wherein the extended virtual method table comprises first and second memory portions, and wherein the edit and continue component provides a reference to the new method in the second memory portion.

44. The system of claim 43, wherein the first and second memory portions of the extended virtual method table are non-contiguous.

45. The system of claim 44, wherein the second memory portion is located so as to maximize a slot index.

46. The system of claim 36, wherein the edit and continue component allows a user to change an existing method in the source code component.

47. The system of claim 46, wherein the edit and continue component allows a user to add a new variable to the existing method.

48. The system of claim 47, wherein the edit and continue component allows a user to add a new variable to the existing method when the existing method is on at least one thread call stack.

49. The system of claim 47, wherein the edit and continue component substitutes edited native code corresponding to the existing method upon a return to the method.

50. The system of claim 49, wherein the edit and continue component determines a return to the method using a breakpoint.

51. The system of claim 46, wherein the edit and continue component allows the user to change an algorithm in the existing method.

52. The system of claim 36, wherein the edit and continue component allows a user to replace an existing method created in a first source language with a new method created in a second source language.

53. The system of claim 36, wherein the edit and continue component compiles the edited source code component, compiles the edited intermediate language component, and executes the edited native code component on an as-needed basis.

54. A system for editing a software program in a common language runtime environment, the software program having a source code component, an intermediate language component, and a native code component, the system comprising:

means for executing a first portion of the native code component;

means for suspending execution of the native code component at a first point;

means for allowing a user to edit the source code component to create an edited source code component;

means for compiling the edited source code component using a source compiler to create an edited intermediate language component;

means for compiling the edited intermediate language component using an intermediate language compiler to create an edited native code component; and

means for executing the edited native code component beginning at the first point.

55. A system for editing a software program having a source code component, and a native code component, the system comprising:

means for executing a first portion of the native code component;

means for suspending execution of the native code component at a first point;

means for allowing a user to edit the source code component to create an edited source code component;

means for converting the edited source code component to create an edited native code component; and

means for executing the edited native code component beginning at the first point;

wherein the means for allowing a user to edit the source code component comprises means for allowing the user to perform one of adding a new field to an object of a class, adding a new method to a class, changing an existing method in the source code component, and replacing an existing method created in a first source language with a new method created in a second source language.

56. A computer-readable medium comprising computer-executable

instructions for:

executing a first portion of a native code component;

suspending execution of the native code component at a first point;

allowing a user to edit a source code component to create an edited source code component;

compiling the edited source code component using a source compiler to create an edited intermediate language component;

compiling the edited intermediate language component using an intermediate language compiler to create an edited native code component; and

executing the edited native code component beginning at the first point.

57. A computer-readable medium comprising computer-executable instructions for:

executing a first portion of a native code component;

suspending execution of the native code component at a first point;

allowing a user to edit a source code component to create an edited source code component;

converting the edited source code component to create an edited native code component; and

executing the edited native code component beginning at the first point;

wherein the computer-executable instructions for allowing a user to edit the source code component comprises computer-executable instructions for allowing the user to perform one of adding a new field to an object of a class, adding a new method to a class, changing an existing method in the source code component, and replacing an existing method created in a first source language with a new method created in a second source language.

58. A runtime system for executing a program in a computer system, comprising:

an edit and continue component having a debugging services interface component interfacing with a debugger application;

wherein the edit and continue component executes a first portion of a native code component, suspends execution of the native code component at a first point, allows a user to edit the source code component using a debugger application to create an edited source code component, converts the edited source code component into an edited native code component, and executes the edited native code component beginning at the first point.

59. The system of claim 58, wherein the edit and continue component converts the edited source code component by compiling the edited source code component using a source compiler to create an edited intermediate language component, and compiling the edited intermediate language component using the intermediate language compiler to create the edited native code component.